TwiRGCN: Temporally Weighted Graph Convolution for Question Answering over Temporal Knowledge Graphs

Aditya Sharma
IISc, Bangalore
adityasharma@iisc.ac.in

Apoorv Saxena
IISc, Bangalore
apoorvsaxena@iisc.ac.in

Chitrank Gupta
IIT Bombay
chigupta2011@gmail.com

Mehran Kazemi
Google Research, Montreal
mehrankazemi@google.com

Partha Talukdar
Google Research, India
partha@google.com

Soumen Chakrabarti
IIT Bombay
soumen@cse.iitb.ac.in

Abstract

Recent years have witnessed interest in Temporal Question Answering over Knowledge Graphs (TKGQA), resulting in the development of multiple methods. However, these are highly engineered, thereby limiting their generalizability, and they do not automatically discover relevant parts of the KG during multi-hop reasoning. Relational graph convolutional networks (RGCN) provide an opportunity to address both of these challenges – we explore this direction in the paper. Specifically, we propose a novel, intuitive and interpretable scheme to modulate the messages passed through a KG edge during convolution based on the relevance of its associated period to the question. We also introduce a gating device to predict if the answer to a complex temporal question is likely to be a KG entity or time and use this prediction to guide our scoring mechanism. We evaluate the resulting system, which we call TwiRGCN, on a recent challenging dataset for multi-hop complex temporal QA called *TimeQuestions*. We show that TwiRGCN significantly outperforms state-of-the-art models on this dataset across diverse question types. Interestingly, TwiRGCN improves accuracy by 9–10 percentage points for the most difficult ordinal and implicit question types.

1 Introduction

Question answering (QA) is a key problem in natural language processing and a long-lasting milestone for artificial intelligence. A large class of approaches for QA makes use of knowledge graphs (KG), which are multi-relational graphs representing facts (KGQA). Temporal KGs (TKG) represent facts that are only valid for specific periods of time as *(subject, relation, object, time range)*, for example, *(Franklin D Roosevelt, position held, President of USA,* $[1933, 1945]$*)*. The problem of answering questions that require temporal reasoning over TKGs (TKGQA) is a special case of KGQA that specifically focuses on the following challenge: temporal questions constrain answers through temporal notions, e.g., "*who was the first president of US during WW2?*" Developing systems for temporal QA is of immense practical importance for many applications. It is considered a more challenging problem than KGQA (Bhutani et al., 2019; Saxena et al., 2020), where questions are typically about persistent, non-temporal facts (e.g., place of birth), with only a small portion of the questions requiring any temporal reasoning (Jia et al., 2018a).

Even though a variety of models have been proposed for the TKGQA recently, they suffer from the following problems: 1) they are either highly engineered toward the task (Jia et al., 2021; Chen et al., 2022) or 2) they do not incorporate graph structure information using Graph Neural Networks (GNN) (Mavromatis et al., 2021; Shang et al., 2022; Saxena et al., 2021). We explore the following hypotheses in this paper: 1) a simple GNN-based solution could generalize better and offer higher performance than highly engineered GNN-based, and TKG embedding-based models; 2) a multi-layer GNN model could do multi-hop reasoning across its layers; 3) not all edges (temporal facts) are equally important for answering temporal questions (see Figure 1), so GNN solutions could benefit from temporally weighted edge convolutions.

Following the aforementioned hypotheses, we develop a novel but architecturally simple TKGQA system that we call "Temporally weighted Relational Graph Convolutional Network" (TwiRGCN). It is based on the Relational Graph Convolutional Network (RGCN) proposed by Schlichtkrull et al. (2018). TwiRGCN introduces a question-dependent edge weighting scheme that modulates convolutional messages passing through a temporal fact edge based on how relevant the time period of that edge is for answering a particular question. In RGCN, convolution messages from all TKG edges are weighted equally. But all edges are not equally important for answering temporal

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2049–2060
May 2-6, 2023 ©2023 Association for Computational Linguistics

questions. For example, in Figure 1, to answer the question "*Who was the first president of the US during WW2?*" the edge with Bill Clinton has little relevance for answering the question. But, regular RGCN would still weigh all edges equally. We address this shortcoming through our proposed modulation. We impose soft temporal constraints on the messages passed during convolution, amplifying messages through edges close to the time period relevant for answering the question while diminishing messages from irrelevant edges. This leads to better, more efficient learning as we are not confusing our model with unnecessary information, as evidenced by our significantly improved performance without the need for any heavy engineering. We explore two different strategies for our convolutional edge weighting, which show complementary strengths. Our experiments establish that TwiRGCN significantly outperforms already strong baselines on *TimeQuestions*. Our contributions are:

- We propose TwiRGCN, a simple and general TKGQA system that computes question-dependent edge weights to modulate RGCN messages, depending on the temporal relevance of the edge to the question.
- We explore two novel and intuitive schemes for imposing soft temporal constraints on the messages passed during convolution, amplifying messages through edges close to the time relevant for answering the question while diminishing messages from irrelevant edges. We also propose an answer-gating mechanism based on the likelihood that the answer is an entity or time.
- Through extensive experiments on a challenging real-world dataset, we find that TwiRGCN substantially outperforms prior art in overall accuracy, and by 9–10% on the implicit and ordinal type questions — categories that require significant temporal reasoning.
- We augment *TimeQuestions* with a TKG and release both code and data at https://github.com/adi-sharma/TwiRGCN.

2 Related Work

Most KGQA systems have focused on answering questions from simple (i.e., 1-hop fact-based questions) (Berant et al., 2013) to multi-hop complex questions requiring multi-fact reasoning (Sun et al., 2019; Saxena et al., 2020). However, only a small fraction of these questions require any temporal reasoning (Jia et al., 2018a). Recent efforts have tried to overcome this gap by proposing models as well as datasets to explicitly focus on temporal reasoning. We review these below.

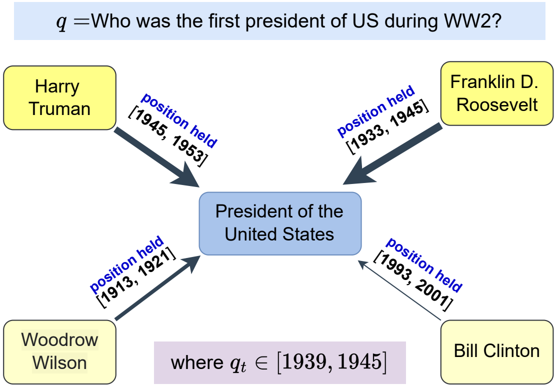

q =Who was the first president of US during WW2?
Harry Truman
position held [1945, 1953]
Franklin D. Roosevelt
position held [1933, 1945]
President of the United States
position held [1913, 1921]
position held [1993, 2001]
Woodrow Wilson
where $q_t \in [1939, 1945]$
Bill Clinton

Figure 1: An illustrative example of how our temporal gating described in Section 4.2 modulated the incoming graph convolution messages for one node depending on the time period of interest for the question. The thickness of an edge here is proportional to the value of the temporal edge weight $m_{tq}^{(e)}$ for that edge. In this example, the entities *Franklin D. Roosevelt* and *Harry Truman*, who were presidents during WW2 $[1939, 1945]$ get the top two highest weights, while *Woodrow Wilson*, who was president during WW1 $[1914, 1918]$ gets a smaller edge weight. In contrast, *Bill Clinton*, whose time period is unrelated to the question, gets a much lower edge weight. Thus, contributing very little to the convolution update of the 'President of the US' node.

Temporal KGQA methods: One line of work uses temporal constraints along with hand-crafted rules to find the answer (Bao et al., 2016; Luo et al., 2018; Jia et al., 2018b). A recent class of models has leveraged advances in TKG embedding methods for answering questions on Temporal KGs. CronKGQA (Saxena et al., 2021) does this by posing a question as a TKG completion problem and finds the answer using the TComplex (Lacroix et al., 2020) score function and BERT (Devlin et al., 2018) question embedding to complete the fact. TempoQR (Mavromatis et al., 2021) uses additional temporal supervision to enrich TKG embeddings, followed by a transformer-based decoder (Vaswani et al., 2017). TSQA (Shang et al., 2022) on the other hand estimate the time in the question and uses it to enrich TKG embeddings for finding the answer. SubGTR (Chen et al., 2022) infers question-relevant temporal constraints using TKG embeddings and applies them as filters to score entities in the question subgraph. Although we, too, use pre-trained TKG embeddings to initialize our generalized RGCN, we use the GNN framework to take advantage of the structural information in the KG in ways that they do not. Recent work (Teru et al., 2020) shows that GNN-based models can encode any logical rule corresponding to a path in the knowledge graph. We refer to this as structural information that shallow embedding-based models cannot access.

RGCN based QA systems: Graph neural networks are increasingly being used in QA systems not specifically meant for temporal reasoning. GraftNet (Sun et al., 2018) uses personalized PageRank to collect a query-relevant subgraph from a global KG, then an RGCN to predict the answer from the relevant subgraph. PullNet (Sun et al., 2019) loops over and expands GraftNet's subgraph to do multi-hop reasoning. EXAQT (Jia et al., 2021) is the system closest to ours: it addresses TKGQA and also uses an RGCN. The RGCN for answer prediction which works on the question subgraph is very similar to that in GraftNet. EXAQT augments it with dictionary matching, heavy engineering, and additional category information. In contrast, TwiRGCN uses a straightforward temporally weighted graph convolution followed by answer gating, as described in Section 4, while still achieving superior performance (see Section 5.3). More details in Section 5.2.

3 Preliminaries

3.1 Temporal Knowledge Graphs (TKG)

KG: Multi-relational graphs with entities (eg: Barack Obama, USA) as nodes and relations r between entities $\{s, o\}$ (e.g., president of) represented as typed edges between nodes. Each edge of this graph, together with endpoint nodes, represents a fact triple $\{s, r, o\}$, e.g., $\{\text{Barack Obama}, \text{president of}, \text{USA}\}$.

TKG: Numerous facts in the world are not perpetually true and are only valid for a certain time period. A TKG represents such a fact as a quadruple of the form $\{s, r, o, [t_{st}, t_{et}]\}$, where t_{st} is the start time and t_{et} is the end time of validity of the fact, e.g., $\{\text{Barack Obama}, \text{president of}, \text{USA}, [2009, 2017]\}$.

3.2 Question Answering on TKGs

Given a question q specified in natural language form and a TKG $\mathcal{G}$, TKGQA is the task of finding the answer to q based on the information that is available (or can be derived) from $\mathcal{G}$. A subgraph of $\mathcal{G}$ is a subset of its nodes with induced edges. In this paper, we assume each question is already associated with a subgraph $\mathcal{G}_q$ relevant to the question. We define $\mathcal{G}_q = (\mathcal{V}_q, \mathcal{R}_q, \mathcal{T}_q, \mathcal{E}_q)$ as the subgraph of $\mathcal{G}$ associated with a question $q \in \mathcal{Q}$, where $\mathcal{Q}$ represents the set of all questions. Each edge $e \in \mathcal{E}_q$ represents a fact $\{v_i, r, v_j, [t_{st}, t_{et}]\}$, where $v_i, v_j \in \mathcal{V}_q$ are entity nodes, $r \in \mathcal{R}_q$ is the relation between them and $t_{st}, t_{et} \in \mathcal{T}_q$ are the start and end times for which the fact is valid.

3.3 Relational Graph Convolutional Networks

Given a KG, each node v_i is initialized to a suitable embedding $h_{v_i}^{(0)}$ at layer 0.Thereafter, Schlichtkrull et al. (2018) propose to update node embeddings $h_{v_i}^{(l+1)}$, at layer $(l+1)$, as follows:

$$h_{v_i}^{(l+1)} = \sigma\left(\sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} \frac{W_r^{(l)} h_{v_j}^{(l)}}{|\mathcal{N}_i^r|} + W_0^{(l)} h_{v_i}^{(l)}\right) \quad (1)$$

where $\mathcal{N}_i^r$ is the set of neighbors of node v_i that are connected via relation edges of type r, $\mathcal{R}$ is the set of relations, $W_r^{(l)}$ are weight matrices associated with each relation type r and layer l. They are initialized using a basis decomposition method.

4 Proposed Method: TwiRGCN

In this section, we develop and describe TwiRGCN ("Temporally Weighted Relational Graph Convolutional Network"), our model for TKGQA.

4.1 Embedding for questions and KG facts

Question embedding: We pass the question text through a pre-trained encoder-only language model (LM) to obtain a question embedding. In particular, we prepend a [CLS] token to the input question and feed it into BERT (Devlin et al., 2019), and then use its output-layer [CLS] embedding as the question embedding q_B. We enable LM fine-tuning during training.

TKG preprocessing for RGCN initialization: We initialize entity and time embeddings using pre-trained TComplEx (Lacroix et al., 2020) embeddings.[1] To obtain these for the *TimeQuestions* dataset (Jia et al., 2021), we first construct a 'background KG' $\mathcal{G} = \bigcup_{q \in \mathcal{Q}} \mathcal{G}_q$ which is the union of all question subgraphs $\mathcal{G}_q$ in the train dataset. As in most temporal KGQA works, we discretize time to a suitable granularity (in our dataset, a year).[2] The graph on which TwiRGCN is run represents every entity as a node v_i and time as edge attribute t_j. Their initial (layer-0) RGCN embeddings $h_{v_i}^{(0)}$ and h_{t_j}, are set to the entity and time embeddings

[1] TComplEx is known to provide high-quality embeddings, but other TKG embedding methods such as TimePlex (Jain et al., 2020) can also be used.

[2] TwiRGCN can be extended to TKGQA datasets that do not provide subgraphs through recently proposed subgraph selection methods (Chen et al., 2022; Shang et al., 2022).

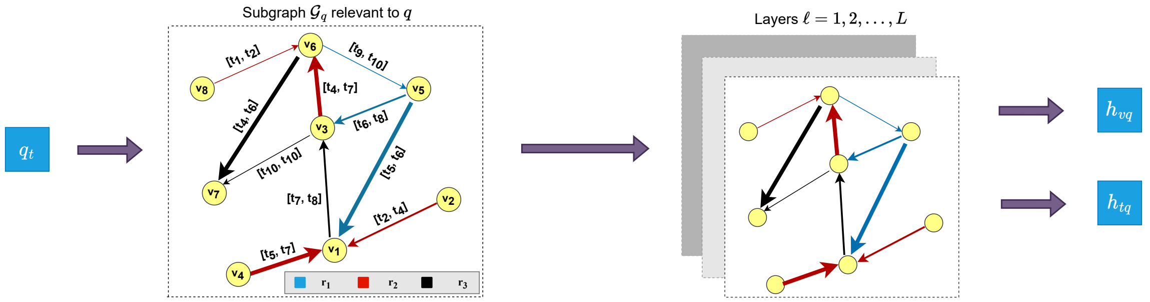


Figure 2: **Left:** Shows temporally weighted convolutional message passing described in Section 4.2 happening across a subgraph G_q for one layer. For the same, we get question-dependent temporal edge weights $m_{tq}^{(e)}$ using *question time*, q_t (described in 4.3). **Right:** As discussed in Section 4.2, embeddings are propagated in the subgraph G_q for a fixed number of layers (L) and hidden units of the final layer are pooled to get entity prediction, h_{vq}. We get time prediction, h_{tq}, by pooling the updated embeddings for all unique times in G_q.

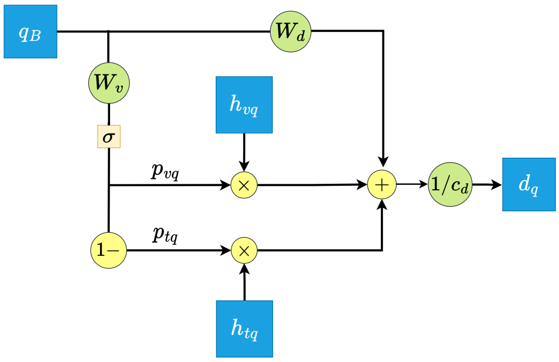


Figure 3: Predicting the answer based on gating entity prediction (h_{vq}) and time prediction (h_{tq}) of a subgraph G_q based on the likelihood that the answer is either an entity (p_{vq}) or time (p_{tq}) given question q, respectively. Details in Section 4.4.

obtained from TComplEx, respectively. We refer to h_{tj} as $h_{st}^{(e)}$ and $h_{et}^{(e)}$ depending on t_j appearing as start or end time for edge e, respectively. When $e = (i, r, j)$, we will use superscript (i, r, j) in place of (e).

4.2 Temporally modulated edge weights

Having available the question subgraph, and the initial entity and time embeddings, our system applies a temporally weighted graph convolution on the local subgraph to enable answering questions that require complex temporal reasoning over a KG. To achieve this, we introduce a question-dependent temporal edge weight $m_{tq}^{(i,r,j)} \in [-1, 1]$ for modulating the convolutional message passed through edge e valid from time $t_{st}^{(i,r,j)}$ to $t_{et}^{(i,r,j)}$ connecting node v_i to v_j via relation r, $\{v_i,\ r,\ v_j,\ [t_{st}^{(i,r,j)}, t_{et}^{(i,r,j)}]\}$ which assigns a weight to that edge depending on how relevant the time period of e is for answering question q. Then, motivated by Eqn. (1), we update the hidden state for a node v_i in the temporal KG at layer $(l+1)$ as:

$$h_{v_i}^{(l+1)} = \sigma\Bigg(W_0^{(l)} h_{v_i}^{(l)} + \sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r} m_{tq}^{(i,r,j)} \frac{W_r^{(l)} h_{v_j}^{(l)}}{|\mathcal{N}_i^r|}\Bigg). \qquad (2)$$

See Figure 1 for an example update for one node.

As shown in Figure 2, after passing messages across a subgraph G_q over L such layers, we pool the hidden states from the final layer of all nodes in G_q to get h_{vq}, the entity prediction. Similarly, we pool the updated embeddings for all unique times in G_q to get h_{tq}, the time prediction. We describe in Section 4.4 how we use h_{vq} and h_{tq} to get the final predicted answer from our model. We use mean pooling in this work, but any other pooling operation can also be used.

4.3 Edge weighting formulations

We explore two different formulations for computing $m_{tq}^{(i,r,j)}$, namely *average* and *interval*, and discuss the motivations behind the two approaches. In Section 5, we empirically show that the inductive bias inherent in each of the two approaches makes them excel at different types of temporal reasoning while giving similar performance overall. We also provide an intuitive explanation of how the edge weighting formulations of the two approaches explain the difference between their empirical results. We first project the question embedding q_B, using a learned projection matrix W_{tq}, to find the *question time* embedding $q_t = W_{tq}q_B$. In the following, $m_{tq}^{(i,r,j)} = m_{tq}^{(e)}$ is the weight for edge e.

4.3.1 TwiRGCN (average)

In this variant, we calculate the edge modulation m_{tq} as the cosine similarity between the question time embedding, and the average of the embeddings for the start and the end time of an edge:

$$m_{tq}^{(e)} = \cos\left(\frac{h_{st}^{(e)} + h_{et}^{(e)}}{2}, q_t\right). \tag{3}$$

This formulation gives a high weight to an edge if the question time falls close to the middle of the time interval for an edge. For example, if the edge times are [2008, 2012] and the question time is 2010, the edge is weighted highly.

4.3.2 TwiRGCN (interval)

In this variant, $m_{tq}^{(e)}$ is defined as the mean of two cosine similarities: (1) the cosine similarity between the start time of the edge and the learned question time embedding, and (2) the cosine similarity between the end time of the edge and the learned question time embedding. Formally,

$$m_{tq}^{(e)} = \frac{\cos(h_{st}^{(e)}, q_t) + \cos(h_{et}^{(e)}, q_t)}{2}. \tag{4}$$

This formulation weighs an edge highly if question time t_q lies within the time interval of the edge.

Generality beyond temporal reasoning While we developed TwiRGCN for temporal reasoning, the edge weighting is more general and could extend to the case where q_t is a "goal" embedding for any goal-directed task.

4.4 Answer type gating

Question answering over TKGs may involve questions whose answer is an entity (e.g., *Who was ...?*) or whose answer is a time (e.g., *When did ...?*). We hypothesize that it should be possible to predict whether the answer to a question is an entity or a time based on the text of the question; making such a prediction helps filter out (or down-weight) a portion of the nodes in that graph that are less likely to be the answer. Toward this hypothesis, we introduce a gating mechanism that learns the likelihood that the answer is an entity p_{vq} or a time p_{tq} given the question:

$$p_{vq} = 1 - p_{tq} = \sigma(w_v q_B), \tag{5}$$

where w_v transforms q_B to a scalar and σ is the sigmoid function that ensures $0 \leq p_{vq} \leq 1$. As shown in Figure 3, we then compute a prediction embedding d_q for question q as a gated sum of the entity prediction and time prediction (see Section 4.2 and Figure 2) added to the question embedding:

$$d_q = \frac{1}{c_d}[p_{vq}h_{vq} + p_{tq}h_{tq} + W_d q_B], \tag{6}$$

where c_d is a constant hyperparameter and W_d is the weight for transforming q_B to the dimension of the entity and time embeddings. Having the prediction embedding d_q, we rank candidate answers (entities and times from the global TKG) based on their similarity to d_q.

Category	Question
Explicit	*Which team won the 2010 F1 world championship?* *What honour did Agatha Christie win in 1971?*
Implicit	*Who did Kevin Garnett play for before Celtics?* *Where was Leonardo Da Vinci when he died?*
Temporal	*What years did the team with fight song Steelers polka win the Superbowl?* *What year did Sam Elliott and Kathryn Ross marry?*
Ordinal	*What was the first satellite to maintain orbit around the earth in space?* *What is the third book of the twilight series?*

Table 1: Examples of questions from each category in *TimeQuestions* dataset, discussed in Section 5.1.

Training We score all possible answer entities and times as a cosine distance with the prediction embedding (d_q), scaled using a constant hyperparameter. We take a softmax over all these scores and train using the cross-entropy loss.

5 Evaluation

5.1 Dataset

Earlier works on TKGQA use the automatically generated CronQuestions dataset (Saxena et al., 2021). A recent analysis, however, shows that this dataset comes with several limitations that stem from its automatic construction method (Chen et al., 2022). Specifically, there are spurious correlations in the dataset that can be exploited by different models to achieve high accuracy (e.g., Mavromatis et al. (2021) report more than 90% accuracy overall and 99% in some categories on this dataset). Therefore, we base our experiments on a recent more challenging dataset, namely *TimeQuestions* (Jia et al., 2021), where the aforementioned models perform poorly (as seen in Table 3).

TimeQuestions has $13.5k$ manually curated questions divided into the train, valid, and test splits containing $7k$, $3.2k$, and $3.2k$ questions, respectively. The questions fall under four types: 'Explicit,' 'Implicit,' 'Temporal,' and 'Ordinal,' based on the type of temporal reasoning required to answer the questions. We show some examples of questions from each of these categories in Table 1. We augment this dataset with question-specific subgraphs generated from WikiData in the final step of the answer graph construction pipeline proposed by Jia et al. (2021). We preprocess all the obtained facts to the *(subject, relation, object, [start*

Dataset	Question
ComQA	*Who played Dumbledore in the 5th harry potter film?*
ComplexWebQuestions	*What is the name of the club the subject of "golden shoes" played for in 2010?*
GraphQuestions	*What sports were in both the 1912 summer Olympics and the 2008 Olympic games?*
LC-QuAD 2.0	*What is the start time for Heidi Klum has spouse as Seal?*
Free917	*What is the price of a 2012 jeep wrangler sport?*

Table 2: Examples of questions requiring temporal reasoning from KGQA datasets (see Section 5.1).

time, end time]) format, and restrict all times to years, a format used by most contemporary TKGs. We create a "background KG” described in Section 4.1 as a union of all subgraphs in the train set. This background KG contains $240k$ facts, $118k$ entities, and 883 relations. We include this augmented *TimeQuestions* dataset and associated code at https://github.com/adi-sharma/TwiRGCN.

Temporal subsets of KGQA datasets: *TimeQuestions* is a compilation of temporal questions from different KGQA datasets. We show the results in Table 4 on a subset of 5 such datasets included in the test set of *TimeQuestions* namely, ComQA (Abujabal et al., 2019), ComplexWebQuestions (Talmor and Berant, 2018), GraphQuestions (Su et al., 2016), LC-QuAD 2.0 (Dubey et al., 2019), and Free917 (Cai and Yates, 2013). Table 2 shows representative examples from these 5 datasets.

5.2 Baseline methods

We compare TwiRGCN against a spectrum of existing methods, including EXAQT, other TKGQA methods, and non-temporal KGQA methods.

Non-temporal KGQA methods: We include Unicorn (Pramanik et al., 2021), which uses Group Steiner Trees for answering questions. We test on two RGCN-based approaches for KGQA, namely, GRAFT-Net (Sun et al., 2018), which attends over relations of neighborhood edges based on the question, and PullNet (Sun et al., 2019), which extends GRAFT-Net for multi-hop questions.

TKGQA methods: We also compare against TKGQA methods CronKGQA (Saxena et al., 2021) and TempoQR (Mavromatis et al., 2021) recently proposed for the CronQuestions dataset. In contrast to TwiRGCN, these do not leverage the powerful GNN framework. CronKGQA frames QA as a KG completion problem to complete the fact the question is interested in, using the TComplex score function and BERT question embedding. TempoQR, on the other hand, enriches pre-trained TKG embeddings with additional supervision from the dataset and uses a transformer (Vaswani et al., 2017) based decoder to predict the final answer.

EXAQT: Jia et al. (2021) propose EXAQT, which is hitherto the best performer on *TimeQuestions*. It is also an RGCN-based TKGQA model that utilizes the GRAFT-Net framework. But in contrast to our model, EXAQT is heavily engineered. It utilizes the ground truth question category information from the dataset at train and test time, so it always knows whether the answer is temporal or belongs to another category. In contrast, our model learns the likelihood that the answer is an entity or time without any explicit supervision through our gating mechanism described in Section 4.4. EXAQT also uses explicit temporal signals from the question, extracted through a dictionary matching-based method using predefined temporal words such as ‘before’, ‘after’, ‘first’, ‘last’, ‘during’, etc. It then enriches its embeddings by utilizing the above in a multi-step end-to-end process. In contrast, our models do not have access to any such information with only a straightforward temporally weighted graph convolution followed by answer gating, as described in Section 4.

5.3 Results

TwiRGCN achieves new state-of-the-art: We compare the accuracy (Hits@1) for different Temporal KGQA models across all question categories found in *TimeQuestions* in Table 3. From this table, we see that our models TwiRGCN (average) and TwiRGCN (interval) achieve significant improvements of up to 3.3% overall absolute accuracy over the previous state-of-the-art model, EXAQT. Additionally, TwiRGCN (average) gets a 9.8% improvement over EXAQT in the ordinal category and TwiRGCN (interval) improves over EXAQT by 9.1% in the implicit category. The questions in both these categories require significant temporal reasoning to find the correct answer. Both models also show a marked improvement of up to 3.4% in the explicit question category.

TwiRGCN (average) vs (interval): Even though the two TwiRGCN variants achieve comparable overall accuracy, they do so in different ways, showing complementary strengths. TwiRGCN (average) achieves a 2.4% improvement over TwiRGCN (interval) in the ordinal category, while TwiRGCN (interval) improves over TwiRGCN (average) for

	Overall	Explicit	Implicit	Temporal	Ordinal
PullNet (Sun et al., 2019)	0.105	0.022	0.081	0.234	0.029
Uniqorn (Pramanik et al., 2021)	0.331	0.318	0.316	0.392	0.202
GRAFT-Net (Sun et al., 2018)	0.452	0.445	0.428	0.515	0.322
CronKGQA (Saxena et al., 2021)	0.462	0.466	0.445	0.511	0.369
TempoQR (Mavromatis et al., 2021)	0.416	0.465	0.36	0.4	0.349
EXAQT (Jia et al., 2021)	0.572	0.568	0.512	0.642	0.42
TwiRGCN *(average)*	**0.605**	**0.602**	0.586	0.641	**0.518**
TwiRGCN *(interval)*	**0.603**	0.599	**0.603**	**0.646**	0.494

Table 3: Comparison of Hits@1 for different Temporal KGQA methods on *TimeQuestions* dataset (Section 5.3). Interestingly, TwiRGCN improves accuracy over SOTA by 3.3% overall and by 9-10% for the most difficult ordinal & implicit question types.

	TwiRGCN	**EXAQT**
ComQA	**0.413**	0.292
ComplexWebQuestions	**0.728**	0.515
GraphQuestions	**0.382**	0.323
LC-QuAD 2.0	0.71	**0.732**
Free917	0	**0.17**

Table 4: Results for EXAQT and TwiRGCN on temporal subsets of well-known KGQA datasets, as discussed in Section 5.3. TwiRGCN beats EXAQT by a high margin up to 21% on ComQA, ComplexWebQuestions, and GraphQuestions, which include questions requiring multi-hop reasoning.

	With gating	**W/o gating**
TwiRGCN (average)	0.605	0.597
TwiRGCN (interval)	0.603	0.597

Table 5: Results of ablation study to see contributions of answer gating described in Section 4.4 on overall Hits@1. We that it contributes about 0.7% on average to the overall accuracy of our models

Temporal Distance from q_t	
Median	**5**
=0	18.3%
≤5	51.5%
≤20	74.8%

Table 6: The median temporal distance from learned q_t to extracted time is just 5 years, while we predict an exact match 18.3% of the time (discussed in Section 5.4).

the implicit and temporal question types. We intuitively explain this behavior as a consequence of their edge weighting function m_{tq}. In TwiRGCN (average), m_{tq} (defined in Eqn. (3)) is at its peak when the average time of edge is close to the question time, enabling it to reason between the temporal ordering of facts more effectively. Thus, helping it better answer ordinal questions of the type *first*, *fourth* or *last occurrence*, etc. In contrast, m_{tq} for TwiRGCN (interval), as defined in Eqn. (4), helps answer questions that require temporal reasoning over specific times rather than just a temporal ordering of facts, which are mainly present in the implicit and temporal categories.

Temporal subsets of KGQA datasets: As mentioned earlier, *TimeQuestions* is a compilation of temporal questions from different KGQA datasets. To provide a finer-grained comparison, we compare TwiRGCN to our most competitive baseline, EXAQT, on these subsets. We show the results in Table 4. TwiRGCN outperforms EXAQT by a high margin of up to 21% in Hits@1 on the *ComQA*, *ComplexWebQuestions*, and *GraphQuestions* datasets. These three datasets contain questions requiring complex multi-hop reasoning. In contrast, we are competitive but perform slightly worse than EXAQT on *LC-QuAD 2.0*, a templated dataset created from SPARQL queries, and *Free917* that primarily consists of quantity-based questions. These results show our model's generalizability and superior performance over our primary baseline on complex multi-hop questions. They also identify a failure mode for questions whose answers are quantities (explored in Section 5.4).

5.4 Analysis

Here we explore how our models behave qualitatively and look at example cases where they perform well and cases where they do not.

Ablation for answer gating: To understand the contribution of the proposed answer gating method described in Section 4.4, we perform an ablation study by removing answer gating from Eqn. (6). By comparing columns of Table 5 we can infer that our proposed answer gating contributes about 0.7% on average to the overall accuracy of our models.

How accurate is predicted question time? We predict a question time embedding q_t close to the time of interest for answering question q, as described in Section 4.3. Here we analyze the effectiveness of this prediction by getting the time with embedding closest to q_t. We then use regex-based time extraction on questions and can extract time for 1199 questions in the test set. As seen in Table 6, out of a time range of 2916 years (including BC and AD years), our median distance from learned question time to extracted time is just 5 years while we predict an exact match 18.3% of the time. Ad-

Ground Truth	Prediction	TwiRGCN (average)		TwiRGCN (interval)	
		with gating	w/o gating	with gating	w/o gating
Entity	Time	3.18 %	7.28 %	3.23 %	4.53 %
Time	Entity	7.99 %	6.88 %	7.3 %	7.82 %

Table 7: Percentage of questions for which answer is an entity but our model incorrectly predicts time and vice versa. We analyze this in Section 5.3 with and w/o answer gating to show that our proposed answer gating helps in reducing such mistakes.

	Hits@1	**Hits@2**	**Hits@3**
EXPLICIT			
EXAQT	0.568	0.602	0.618
TwiRGCN	**0.602**	**0.618**	**0.628**
IMPLICIT			
EXAQT	0.512	0.575	0.612
TwiRGCN	**0.603**	**0.622**	**0.637**
ORDINAL			
EXAQT	0.42	0.47	0.49
TwiRGCN	**0.518**	**0.542**	**0.553**

Table 8: Effects of increasing k for Hits@k. As discussed in Section 5.4, TwiRGCN significantly outperforms EXAQT across categories of questions even as k is increased.

Temporal (Hits@1)	**±0**	**±1**	**±3**
EXAQT	0.642	0.653	0.667
TwiRGCN (average)	0.641	0.649	0.671
TwiRGCN (interval)	**0.646**	**0.659**	**0.682**

Table 9: Effects of increasing the answer temporal window on model performance for Temporal type questions. As discussed in Section 5.4, TwiRGCN (interval) gets even more accurate relative to EXAQT as we increase the temporal tolerance window.

ditionally, for 51.5% of the questions the distance is $\leq$5 years, while it is $\leq$20 years for $\sim$75% of the questions. Our simple-to-learn q_t which is just a linear transform of q_B works reasonably well. Better q_t would result in even more performance improvements. We leave that for future work.

Dominant errors: We do an error analysis over quantity-type questions, a challenging query class. Neither EXAQT nor TwiRGCN perform well on quantity-type questions. Out of a total of 224 quantity questions, EXAQT gets 0.1% accuracy while TwiRGCN gets 0.05%. This is because current TKGQA models treat quantities such as "2.55" or "16,233" as independent entities, instead of scalar numeric values. Additionally, from Section 5.3, we reconfirm that current TKGQA models fail on this bucket, so future work can direct special attention here. Examples: "*What was Panama's fertility rate in 2006?*" A: 2.55; "*What was the population of Bogota in 1775?*" A: 16,233.

Reducing answer type mistakes: In this study, we estimate TwiRGCN's propensity to make answer type mistakes. We define these mistakes as questions where the answer was an entity, but our model predicted a time or vice versa. From Table 7 we see that our answer gating mechanism mentioned in Eqn. (5) helps reduce such mistakes. For TwiRGCN (average), gating cuts entity-to-time mistakes by more than half.

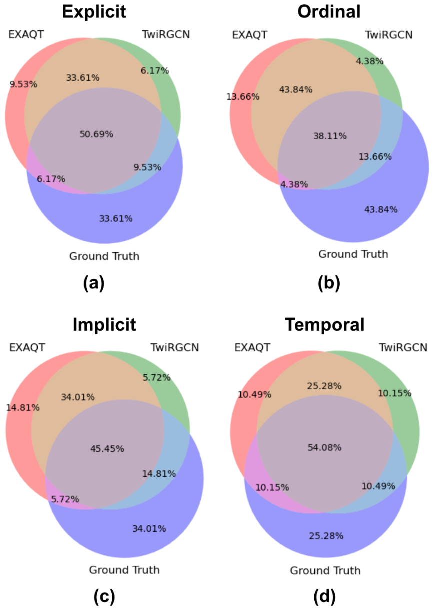


Figure 4: Venn diagrams for the prediction overlap of EXAQT, ground truth and our best model for each category. As described in Section 5.4, for Explicit, Implicit, and Ordinal question types TwiRGCN gives the right answers for most questions that EXAQT answers correctly, while correctly answering a much larger set that EXAQT gets wrong.

Increasing k for Hits@k: We extend the analysis in Table 3 increasing k from 1 to 3 for Hits@k on *TimeQuestions*. From Table 8 we see that the performance of TwiRGCN is robust to increasing k. It significantly outperforms EXAQT across categories even as k is increased for Hits@k.

Increasing temporal tolerance window: In Table 9, we explore the effects of increasing the time window for marking an answer correct for temporal questions. This means if the ground truth answer is 1992, and the predicted answer is 1990 for a question, it will be marked as incorrect in the $\pm$1 column and correct in the $\pm$3 column. We find that our model, specifically TwiRGCN (interval) gets even more accurate relative to EXAQT as we increase the temporal tolerance window. This implies that TwiRGCN is robust at ranking gold answers high up, even if they do not achieve rank 1.

Prediction overlap: We study the overlap of predictions between EXAQT, TwiRGCN, and ground truth. As seen in Figure 4, for Explicit, Implicit, and Ordinal question types our model gives the right answers for most questions that EXAQT answers correctly (missing less than 6% on average), while correctly answering a much larger set that EXAQT gets wrong. This split is more even between the two models for the temporal-type questions.

6 Conclusion

In this paper, we proposed TwiRGCN, a TKGQA system that employs a novel, temporally weighted graph convolution for answering questions that require complex temporal reasoning over a TKG. TwiRGCN modulates the convolutional messages through a TKG edge based on the relevance of the edge time interval to the question. We present two temporal weighting schemes with complementary strengths, intuitively explained through their simple formulations. We also propose an answer gating system for incorporating the pooled entity and time embeddings from TwiRGCN in the prediction, based on the likelihood that the answer is a time or an entity, given the question. Despite its relative simplicity, TwiRGCN gives significantly superior TKGQA accuracy on a challenging dataset compared to more heavily engineered baselines.

Acknowledgements

We thank Jeremy Cole and Srini Narayanan for their valuable feedback. We also thank the anonymous reviewers for their constructive comments. Soumen Chakrabarti was supported in part by grants from Amazon, Google, IBM, and SERB.

7 Limitations

TwiRGCN is limited by the need for relevant subgraphs for each question to be provided in the dataset. Such subgraphs have been provided in the *TimeQuestions* dataset used in the current work, but that may not be true for all TKGQA datasets. This limitation may be addressed for datasets that do not provide subgraphs through recently proposed subgraph selection methods (Chen et al., 2022; Shang et al., 2022; Jia et al., 2021), but we leave that exploration for future work.

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. ComQA: A community-sourced dataset for complex factoid question answering with paraphrase clusters. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 307–317, Minneapolis, Minnesota. Association for Computational Linguistics.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514, Osaka, Japan. The COLING 2016 Organizing Committee.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Nikita Bhutani, Xinyi Zheng, and HV Jagadish. 2019. Learning to answer complex questions over knowledge bases with query composition. In *CIKM*.

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 423–433.

Ziyang Chen, Xiang Zhao, Jinzhi Liao, Xinyi Li, and Evangelos Kanoulas. 2022. Temporal knowledge graph question answering via subgraph reasoning. *Knowledge-Based Systems*, page 109134.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II*, pages 69–78.

Prachi Jain, Sushant Rathi, Soumen Chakrabarti, et al. 2020. Temporal knowledge base completion: New algorithms and evaluation protocols. *arXiv preprint arXiv:2005.05035*.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018a. Tempquestions: A benchmark for temporal question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1057–1062, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018b. Tequila: Temporal question answering over knowledge bases. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 1807–1810, New York, NY, USA. Association for Computing Machinery.

Zhen Jia, Soumajit Pramanik, Rishiraj Saha Roy, and Gerhard Weikum. 2021. Complex temporal question answering on knowledge graphs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 792–802.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *International Conference on Learning Representations*.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194, Brussels, Belgium. Association for Computational Linguistics.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Soji Adeshina, Phillip R. Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2021. Tempoqr: Temporal question reasoning over knowledge graphs.

Soumajit Pramanik, Jesujoba Alabi, Rishiraj Saha Roy, and Gerhard Weikum. 2021. UNIQORN: Unified Question Answering over RDF Knowledge Graphs and Natural Language Text. In *arXiv*.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. 2021. Question answering over temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6663–6676, Online. Association for Computational Linguistics.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Chao Shang, Guangtao Wang, Peng Qi, and Jing Huang. 2022. Improving time sensitivity for question answering over temporal knowledge graphs. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8017–8026, Dublin, Ireland. Association for Computational Linguistics.

Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gür, Zenghui Yan, and Xifeng Yan. 2016. On generating characteristic-rich question sets for qa evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 562–572.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *EMNLP-IJCNLP*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *EMNLP*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

Komal Teru, Etienne Denis, and Will Hamilton. 2020. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pages 9448–9457. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

TwiRGCN: Temporally Weighted Graph Convolution for Question Answering over Temporal Knowledge Graphs
(Appendix)

A Additional Analyses

A.1 Complete prediction overlap

In Figure 5, we extend our analysis in 5.4 by providing the complete prediction overlap for both our models with EXAQT and ground truth across all question categories in *TimeQuestions*.

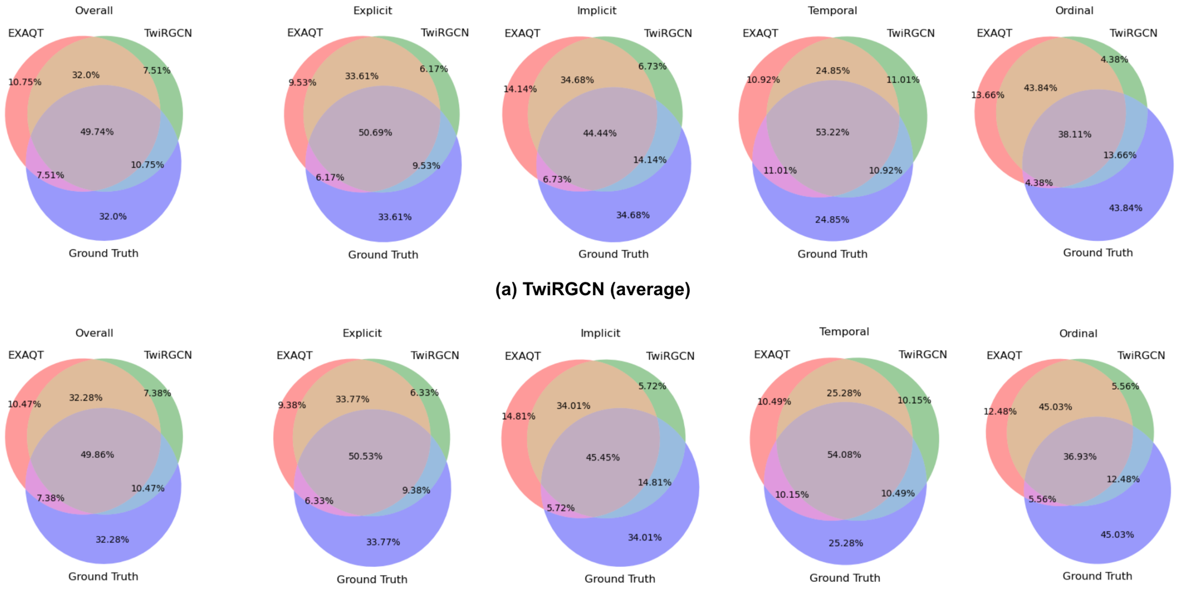
B Hyperprameters

We use the following hyperparameters:

- Number of layers, $L = 2$
- $c_d = 3$
- train batch size = 32
- valid batch size = 5
- LR = 0.00004
- Decay for LR = 0.4 every 10 epochs
- Cosine distance scaling constant for training (described in Section 4) = 30

Model and program execution details:

- Number of parameters = 2,223,833
- 11GB Nvidia GPU used with cudatoolkit 11.1
- Time per training epoch = 1:04 min
- Number of epochs to convergence on average = 50
- Early stopping used and implemented in code with patience = 10
- Validation overall Hits@1 for TwiRGCN (average) = 0.606
- Validation overall Hits@1 for TwiRGCN (intervall) = 0.602
- Performance is fairly stable around current hyperparameters without much tuning, except for LR decay rate. We used around 5–7 training runs with different decay settings to get the current rate. TwiRGCN is stable around current settings.
- Hyperparameters were tuned by manually inspecting loss behavior. Final values were selected based on a sustained, stable good performance on the test set for 3 runs.



(a) TwiRGCN (average)

(b) TwiRGCN (interval)

Figure 5: Venn diagrams for the prediction overlap of EXAQT, ground truth, and our two models TwiRGCN (average) in (a) and TwiRGCN (interval) in (b), as discussed in Appendix A.1.